OPTIMAL DESIGN OF PIPE NETWORKS BY GENETIC ALGORITHMS

Sérgio Augusto Araújo da Gama Cerqueira, sergioc@ufsj.edu.br
Rodrigo Ferri Dias, rodrigoferri@axtelecom.com.br
Departamento de Engenharia Mecânica - UFSJ

Ricardo Hiroshi Caldeira Takahashi, taka@mat.ufmg.br
Departamento de Matemática - UFMG

Abstract. *In this work a new genetic algorithm is proposed to tackle the problem of selecting the optimal diameters of the pipes in a network. When a limited available set of standard pipe diameters is available, the problem at hand becomes a hard non-linear discrete optimization problem. Even for relatively small systems, such as the standard literature one taken here as an application example, the sheer number of combinations rules out the enumeration of solutions. The genetic algorithm proposed, uses integer encoding, presents new crossover and mutation operators and employs a two-step selection operator, combining the two most common methods, roulette and tournament, and considering both fitness and feasibility. The algorithm is shown to be both robust and efficient, consistently finding solutions equal to the best yet found for a classical test problem, with a reduced number of calls to the network simulator.*

Keywords: *pipe networks, genetic algorithm, nonlinear discrete optimization*

1. INTRODUCTION

Finding a needle in a haystack. That is the laymen expression for a hard, tedious operation. It pales, though, in comparison to the task of finding the global optimum of even a simple discrete, nonlinear problem. Heavy investments are made everyday in the design of new or the upgrade of existing pipe networks, carrying water, oil, or natural gas. It is the task of the engineer to find better designs, with reduced costs, coping with the demands and ensuring the required minimum pressures.

Over the last 15 years several authors have applied evolutionary methodologies to the optimization of pipe networks. The genetic algorithm, in its original binary coded version, was first applied by Savic and Walters (1995); Dandy *et al.* (1996); Savic and Walters (1997). More recently, others have proposed new developments of the method (Walters *et al.*, 1999; Wu *et al.*, 2001; Vairavamoorthy and Ali, 2005; Neelakantan and Suribabu, 2005) and used different encodings (Vairavamoorthy and Ali, 2000).

In this work, an integer coded genetic algorithm is developed and applied to a benchmark water distribution problem. New crossover and mutation operators are proposed, inspired by those employed in real coded GAs. Feasibility, an often neglected problem, is tackled by a variation of the procedure proposed by Deb (2000), a two-step selection operator, which combines the two most common methods, roulette and tournament.

2. THE PIPE NETWORK DESIGN PROBLEM

The optimal selection of pipe diameters in a network is formulated as follows (Dandy *et al.*, 1996). For a given network topology and a set of specified demand patterns at the nodes, one wish to find the combination of diameters that minimizes the total cost, subject to the following restrictions.

Find

$$\mathbf{D}^* : \arg\min C(\mathbf{D}) \tag{1}$$

subject to

$$\forall i \in \{1,\ldots,n\} \qquad \sum_{j=1}^{m} a_{ij} Q_j - q_i = 0 \tag{2}$$

$$\forall j \in \{1,\ldots,m\} \qquad \sum_{i=1}^{n} a_{ij} H_i = h_j \tag{3}$$

$$\forall i \in \{1,\ldots,n\} \qquad H_i^L \leq H_i \leq H_i^U \tag{4}$$

$$\forall j \in \{1,\ldots,m\} \qquad D_j \in \{d_1,\ldots,d_{N_D}\} \tag{5}$$

The first restriction (Eq. 2) represents the conservation of mass, which must be kept at each one of the n nodes, q_i being the demand at the node i, and a_{ij} are the elements of the $\mathbf{A}$ matrix ($n \times m$), which identifies the pipes that connect to a certain node. The second restriction (Eq. 3, the head loss h_j in each of the m pipes is a known function of the flow in the

pipe (Q_j), its diameter (D_j), length (L_j) and hydraulic properties. The third restriction (Eq. 4) imposes a maximum and a minimum pressure (H_i) limitations at the nodes. Finally, it is imposed (Eq. 5) that the diameters belong to a N_D-sized specified set.

3. GENETIC ALGORITHMS

The denomination genetic algorithms (GAs) is employed to designate optimization algorithms that perform an approximate global search, characterized by (Carrano *et al.*, 2006): i) the reliance on the information obtained by the evaluation of several points in the search space. Each "current point" is called an individual, and the set of individuals is called the population. The algorithm keeps a population, instead of keeping a single "current point," as would be the case in most optimization algorithms. ii) The population converges to a problem optimum through sequential applications, at each iteration, of the following genetic operators.

- Mutation – Some individuals are randomly modified, in order to reach other points of the search space.
- Crossover – The individuals, randomly organized pairwise, have their space locations combined, in such a way that each former pair of individuals gives rise to a new one.
- Selection – The individuals, after mutation and crossover, are evaluated. They are chosen or not chosen for being inserted in the new population through a probabilistic rule that gives a greater probability of selection to the "better" individuals (the ones with smaller objective function evaluation).
- Elitism – A subset of the former population that contains the "best" individuals is deterministically inserted in the new population.

Each operator can be implemented in several alternative ways: a combination of specific realizations of these operators constitutes an instance of GA. Besides these basic genetic operators, there are others which are employed in some GAs.

4. THE INTEGER CODED GENETIC ALGORITHM

The individuals in the integer coded genetic algorithm are represented by m-sized vectors of integer numbers, each one representing the diameter of one pipe. The population of N individuals is therefore a $N \times m$ matrix. Most genetic algorithms employ *binary encoding*, in which the individuals consist of a long string of zeros and ones, with sub-strings associated with the pipe diameter. The integer encoding brings the problem closer to its physical reality, in which the type 4 pipe is similar to the 3 and the 5 types of pipe. This property is used in the crossover and mutation operators. The initial population was randomly generated, using a uniform distribution. The algorithm was implemented in *m*-language, compatible with Matlab and GNU/Octave.

4.1 Selection

Survival of the fittest, the lemma of evolution, is the principle by which individuals are either selected for mating or forgotten. A linear relation was used to determine the fitness function $(f^{\$})$ from the individual's objective function:

$$f^{\$} = \frac{\max(C) - C_i}{\max(C) - \min(C)} \tag{6}$$

in which C is the vector of costs and C_i is the cost of the *i*-th individual.

The two most common procedures for selection are the *roulette* and the *tournament*. In the *roulette*, the probability of selection of an individual is made proportional to its relative fitness, the relative fitness of an individual is the fitness divided by the sum of the fitness of all the individuals.

In the *tournament*, two individuals are selected from the population and the winner is the one with the greatest fitness. The number of tournaments is therefore equal to the size of the population. Deb (2000) proposed a variation of the tournament that incorporates the treatment of feasibility, as follows.

1. If both individuals are feasible, decision by the fitness.
2. If at least one of the individuals is infeasible, decision by feasibility.

The feasibility (f^p) was measured by the number of times that the required head at a node was not achieved $(n_{p,i})$, divided by the number of nodes (n).

$$f^p = 1 - \frac{n_{p,i}}{n} \tag{7}$$

In this work, in order to reduce the number of infeasible individuals, both procedures where employed in succession. If the entire population was feasible, then a roulette was employed to sample the population, leaning into the exclusion the very worst individuals. If one or more infeasible individuals existed in the population, the sample was based on feasibility, leaning into the exclusion the most unfeasible individuals.

In a second step, tournament was employed to further select the population towards low cost and feasible individuals. The number of individuals selected is equal to the size of the population.

4.2 Crossover

From two parents (p_1 and p_2) extracted from the selected population, two descendants are produced. If a randomly chosen variable is greater than the crossover probability ($p_c = 0.95$), the descendants are made equal to the parents; otherwise, crossover occurs.

A single crossover operator was employed, inspired by real coded genetic operators. The elements of the descendants are generated by the rounded linear combination of the corresponding elements in two parents,

$$d_j = \text{round}(p_{1,j} + (1 - \alpha)p_{2,j}) \tag{8}$$

where $\alpha = U(0 - \gamma, 1 + \gamma)$ is a uniform random variable, in the interval $(0 - \gamma, 1 + \gamma)$, where $\gamma = 0.2$. A check is made after the crossover operation for out of bounds elements, *i.e.* if $0 \leq d_j \leq N_D$. If that is true, the element is brought back into the range by making it equal to nearest extreme value.

4.3 Mutation

A point to point mutation operator was used, meaning that if a randomly chosen variable is greater than the mutation probability ($p_m = 0.07$), the value of a component of the population matrix is altered,

$$d_j = \text{round}(d_j + \beta\Delta) \tag{9}$$

where $\beta = U(-1, 1)$ is a uniform random variable, and $\Delta = 3$. Also here, a check is made for out of bounds elements, and the same fix as in crossover is employed.

4.4 Elitism

Only the best individual yet found is kept in a list apart. If it does not appear in the generated population, it is inserted in place of the worst individual present. We refrained from using larger sets to avoided the loss of diversity in the population, which is known to cause premature convergence.

4.5 The Network Simulator

The network was simulated by the gradient method, proposed by Todini and Pilati (1987), and described in details by Lansey and Mays (2000). It was selected for its good convergence characteristics and by the reduced computational costs (Rossman, 2000a).

5. TEST PROBLEM: THE NEW YORK CITY WATER SUPPLY TUNNELS

The expansion of the New York city water supply tunnels network was first tackled by Schaake and Lai in 1969 (Dandy *et al.*, 1996). Over the years, it has become a benchmark problem, studied by a large number of authors. By the time of the original paper, the system was composed of a network of deep rock tunnels of large diameters (up to 204 inches), shown in Fig. 1. Due to the original problem definition, and to allow for comparison with previous results, US customary units are used throughout this work. It was proposed to expand the system by adding a series of parallel tunnels to the existing ones – a decision was later taken to build a third tunnel, which is still under construction. The system has two main tunnels, running south from the Hillview reservoir, which is 300 ft above all the other nodes. The demands and minimum total head requirements for each node of the network are presented in Tab. 1. It was found (Dandy *et al.*, 1996) that, given the increase in demand, nodes 16, 17, 18, 19, and 20 would fall significantly bellow the minimum requirement.

The lengths of the arches and the diameters of the existing pipes are presented in Tab. 2. In the original work, the head loss was determined by the Hazen-Williams equation, with a roughness coefficient $c = 100$ for all pipes. There are variations in the constants in the Hazen-Williams, depending on the source. It is written bellow as used in this work, in the original paper, and in EPANET2 (Rossman, 2000b), a pipe network simulator developed by the USA's Environmental Protection Agency.

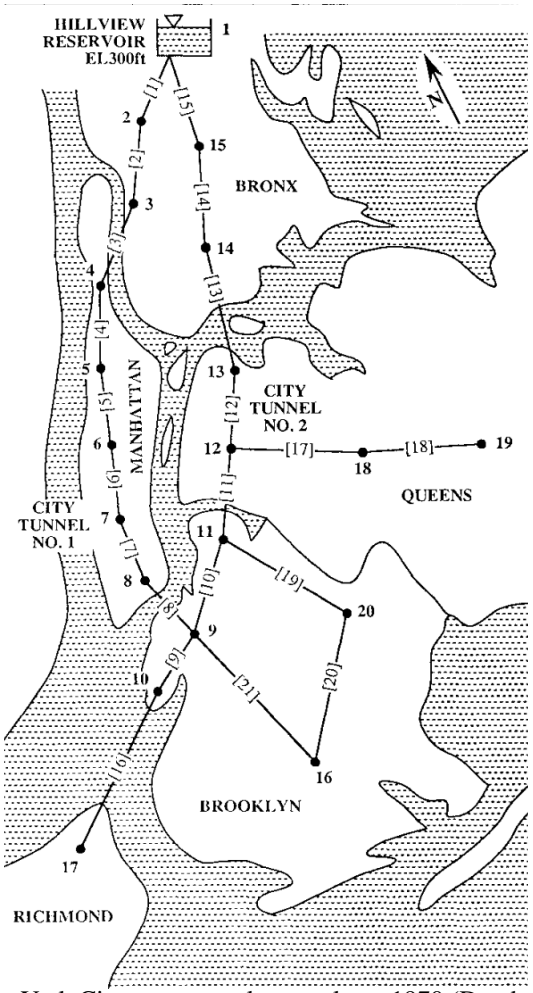

Figure 1. New York City water supply tunnels, c. 1970 (Dandy *et al.*, 1996).

$$h_f = 4.729L\left(\frac{Q}{c}\right)^{1.852} D^{-4.8704} \tag{10}$$

It should be noted that the Eq. 10 is extremely sensitive to the constants (in special the exponents) and therefore some of the best results found in the literature, *e.g.* Savic and Walters (1997); Cunha and Ribeiro (2004), are not feasible if the relation is used as presented above.

There are 15 available discrete pipe diameters (36, 48, 60, 72, 84, 96, 108, 120, 132, 144, 156, 168, 180, 192, 204 inches) and one extra possible decision which is the "do nothing" option. Only the costs of acquisition and installation of the pipes are considered. The objective function becomes the sum of the costs of the added tubes,

$$C_i = 1.1\sum_{j=1}^{m} D_{i,j}^{1.24} L_j \tag{11}$$

where C_i is the cost of the i-th network, $D_{i,j}$ is the diameter of the pipe used in the *j*-th arch and L_j is the length of that arch.

Some of the best results found in the literature are presented in Tab. 3. Schaake and Lai's solution (Dandy *et al.*, 1996) was obtained by a linear programming approach, in which the decision variables, the pipe diameters, were linearized by being raised to the power 2.63, and the total heads at each node were previously fixed. It lead to a costly design, at US$78.09 million, in which nearly all pipes were duplicated, using non-standard diameters.

More than a decade later, Gessler (1982), used a partial enumeration of the discrete pipe diameters in either of the main tunnels to initialize a continuous gradient search. His best solution required the duplication of seven pipes in tunnel number 1, at a cost of US$ 41.8 million.

Table 1. Nodal data for the New York City water supply tunnels

Node	Demand $(\mathrm{ft}^3/\mathrm{s})$	Minimum total head (ft)
1	Reservoir	300
2	92.4	255
3	92.4	255
4	88.2	255
5	88.2	255
6	88.2	255
7	88.2	255
8	88.2	255
9	170.0	255
10	1.0	255
11	170.0	255
12	117.1	255
13	117.1	255
14	92.4	255
15	92.4	255
16	170.0	260
17	57.5	272.8
18	117.1	255
19	117.1	255
20	170.0	255

Table 2. Network data for the New York City water supply tunnels

Arch	Start node	End node	Length (ft)	Existing diameter (in)
1	1	2	11600	180
2	2	3	19800	180
3	3	4	7300	180
4	4	5	8300	180
5	5	6	8600	180
6	6	7	19100	180
7	7	8	9600	132
8	8	9	12500	132
9	9	10	9600	180
10	9	11	11200	204
11	11	12	14500	204
12	12	13	12200	204
13	13	14	24100	204
14	14	15	21100	204
15	15	1	15500	204
16	10	17	26400	72
17	12	18	31200	72
18	18	19	24000	60
19	11	20	14400	60
20	20	16	38400	60
21	16	9	26400	72

Table 3. A comparison of some of the best designs for the New York Tunnels

Pipe	Schaade and Li (1969)	Gessler(1982)	Dandy *et al.* (1996)	Savic and Walters (1997)[1]
1	52.02	0	0	0
2	49.9	0	0	0
3	63.41	0	0	0
4	55.59	0	0	0
5	57.25	0	0	0
6	59.19	0	0	0
7	59.06	100	0	108
8	54.95	100	0	0
9	0	0	0	0
10	0	0	0	0
11	116.21	0	0	0
12	125.25	0	0	0
13	126.87	0	0	0
14	133.07	0	0	0
15	126.52	0	120	0
16	19.52	100	84	96
17	91.83	100	96	96
18	72.76	80	84	84
19	72.61	60	72	72
20	0	0	0	0
21	54.82	80	72	72
Cost US$ million	78.09	41.8	38.81	37.13

The application of genetic algorithms led to better results in the 1990's. Dandy *et al.* (1996) used a binary genetic algorithm with Gray's encoding. This leads to better locality characteristics, as in this code the next string is only a bit distant from the other. The solution they found, at a cost of US$ 38.81 million, is the best known feasible solution to the problem.

Savic and Walters (1997) noticed the sensitivity of the Hazen-Williams equation to the exponents. With coefficients that are different from the ones used here, he found a solution which is slightly infeasible, but, at a cost of US$ 37.13 million, significantly cheaper.

6. RESULTS

The experiments were run on an Intel *Core 2 Quad* GNU/Linux workstation, using Matlab. Each execution took several minutes, scaling almost linearly with the population size, and the full experiment took a little under three days, using the four processors. The results of the simulations are shown in Tab. 4 and Fig. 2. The horizontal bars in the figure show the interval between the best and the worst values of the objective function, for a given population size, over 105 runs of the algorithm. The vertical bars show the range of the number of calls of the objective function and the network simulator, a proxy for computational cost, until the best result of the run was found. It should be remarked that this number is raw, as no effort was made to avoid the re-evaluation of the objective function and/or the repetition of the simulation of an individual.

Table 4. Statistical data for objective function in the experiments

Population	Worst	P10	P25	P50	P75	P90	Best	Mean
100	45.573	40.295	40.193	39.957	39.598	38.814	38.814	39.899
200	41.424	40.222	40.176	39.699	39.598	38.814	38.814	39.716
300	40.265	40.193	40.176	39.850	39.674	38.814	38.814	39.773
400	40.193	40.176	40.176	39.699	39.598	38.814	38.814	39.716
500	40.193	40.176	40.176	39.810	39.699	39.296	38.814	39.814

Over the course of the whole experiment the same best configuration was found, for all the population sizes tested, with a cost of US$38.814 million – the same found by Dandy *et al.* (1996). On the other hand, there were important differences in the worst configurations, which improved with the size of the population from well over US$45 million, with

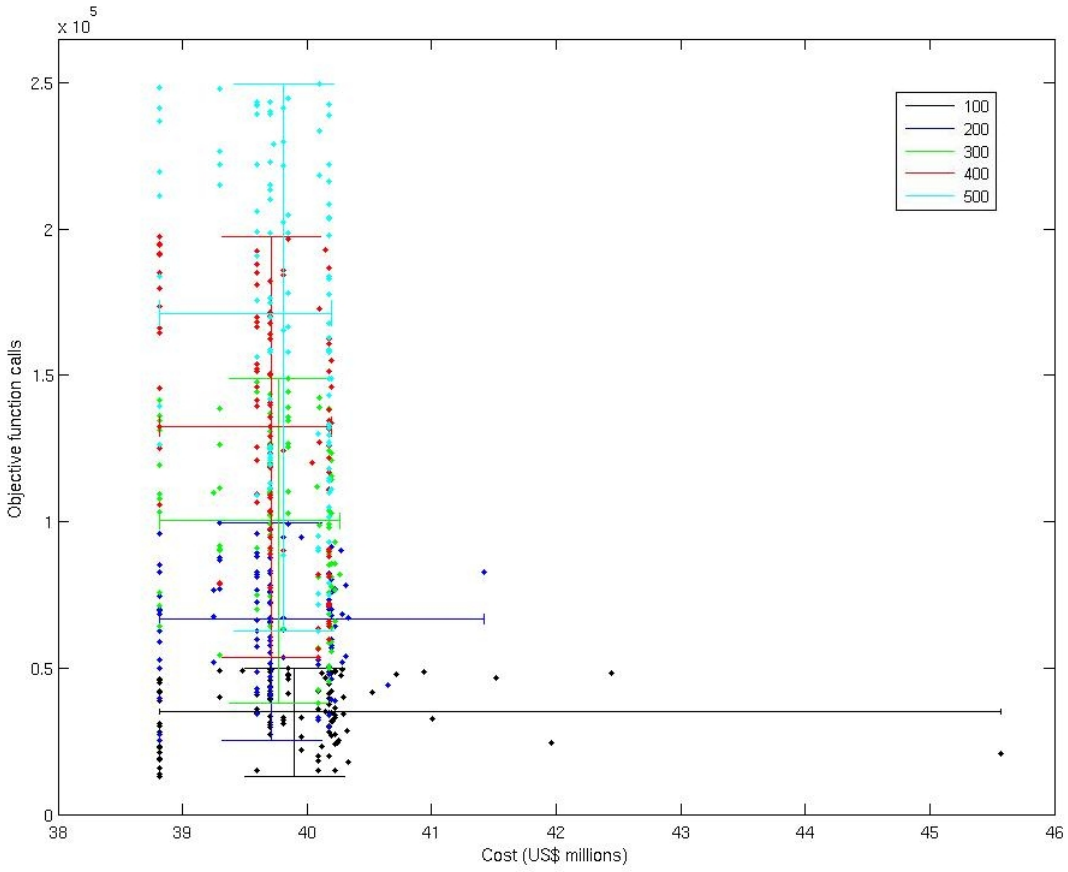

Figure 2. Objective function value (cost) and number of calls of objective function for various population sizes (error bars envelop the best and worse results of 105 runs of the algorithm)

the smallest population of 100 individuals, to US$40.193 million, with the largest (500 individuals). This improvement, though, came at a cost of an almost fivefold increase in the mean computational cost, as measured by the mean number of calls to the objective function, until the best individual was found.

The mean value of the objective function, for each population size, had a more erratic behaviour. It improved from US$ 39.90 to US$39.72, when the population size grew from 100 to 200. From then on, the increase in the size of the population cannot be associated with a decrease in the mean value of the objective function. Most remarkably, the mean value found for the largest population size lies roughly in the middle of those of the two smallest sizes. That can be attributed to the greatest "inertia" of the largest populations, that is, the time it takes for an improved trace to spread throughout the population.

The P10 percentile confirms the graphical impression that most of the differences among the results are in the worst cases and the best cases. Smaller sizes will generate a larger number of failed runs, while the largest population will often fail to achieve the better ones. In fact, the smallest population (100 individuals) was the best at finding the best solution – it was able to find it in 19 of the 105 (18,1%) runs. The largest population (500 individuals), meanwhile, only found it in 8 (7.6%) runs.

On computational cost, it is difficult to compare the algorithm, as most authors fail to provide information on the number of experiments and calls to the objective function. Dandy *et al.* (1996) used 200,000 evaluations in his GAs, with populations ranging from 100 to 500 individuals. The GA here presented when it found the minimal cost solution had a mean number of calls to the objective function of 99.7×10^3, with the minimum of 12.8×10^3 and a maximum of 248.5×10^3 calls.

7. CONCLUSIONS AND FUTURE WORK

The integer coded genetic algorithm was shown to be both robust and efficient, consistently finding solutions equal to the best yet found for a classical test problem, with a reduced number of calls to the network simulator. It employs a

unique selection operator, combined with new mutation and crossover operators.

As is usual in restricted optimization problems, the best solutions are found near the boundary between the feasible and the infeasible regions of the variables space. The results found with the new two-step selection operator have inspired a systematic study of its behaviour, which is currently under way. Also in development are new operators, making use of a recently developed metric for network distances. It is hoped that those will reduce even further the computational cost of the algorithm.

Though the application problem was a water distribution pipe network, the use of the algorithm is not restricted – as it is, it can be used for the optimization of pipe networks for any incompressible fluids. A change in the network simulator would allow for the optimization of gas networks; other modifications would allow for the inclusion of pumps or compressors.

8. ACKNOWLEDGEMENTS

The second author would like to thank FAPEMIG for his scholarship.

9. REFERENCES

Carrano, E., Soares, L. *et al.*, 2006. "Electric distribution network multiobjective design using a problem-specific genetic algorithm". Power Delivery, IEEE Transactions on, vol. 21, no. 2, pp. 995– 1005.

Cunha, M. and Ribeiro, L., 2004. "Tabu Search Algorithms for Water Network Optimization". European Journal of Operational Research, vol. 157, pp. 746–758.

Dandy, G. C., Simpson, A. R. *et al.*, 1996. "An Improved Genetic Algorithm for Pipe Network Optimization". Water Resources Research, vol. 32, pp. 449–458.

Deb, K., 2000. "An Efficient Constraint Handling Method for Genetic Algorithms". Computer methods in applied mechanics and engineering, vol. 186, no. 2-4, pp. 311–338.

Gessler, J., 1982. "Optimization of pipe networks". In "Proc. Inst. on Urban Hydr. and Sediment Control", University of Kentucky, Lexington, KY, pp. 165–171.

Lansey, K. E. and Mays, L. W., 2000. Hydraulics of Water Distribution Systems, chap. 4. McGraw-Hill, New York.

Neelakantan, T. R. and Suribabu, C. R., 2005. "Optimal Design of Water Distribution Networks by a Modified Genetic Algorithm". International Journal of Civil and Environmental Engineering, vol. 1, no. 1, pp. 20–34.

Rossman, L. A., 2000a. Computer Models/EPANET, chap. 7. McGraw-Hill, New York.

Rossman, L. A., 2000b. EPANET 2 Users Manual. Cincinnati, OH, USA.

Savic, D. A. and Walters, G. A., 1995. "Genetic Operators and Constraint Handling for Pipe Network Optimization". In "Evolutionary Computing, AISB Workshop", pp. 154–165.

Savic, D. A. and Walters, G. A., 1997. "Genetic Algorithms for Least-Cost Design of Water Distribution Networks". Journal of Water Resources Planning and Management, pp. 67–77.

Todini, E. and Pilati, S., 1987. "A gradient method for the analysis of pipe networks". In "International Conference on Computer Applications for Water Supply and Distribution", Leicester Polytechnic, UK, pp. 1–20.

Vairavamoorthy, K. and Ali, M., 2000. "Optimal design of water distribution systems using genetic algorithms". Computer-Aided Civil and Infrastructure Engineering, vol. 15, no. 5, pp. 374–382.

Vairavamoorthy, K. and Ali, M., 2005. "Pipe index vector: A method to improve genetic-algorithm-based pipe optimization". Journal of Hydraulic Engineering, vol. 131, no. 12, pp. 1117–1125.

Walters, G. A., Halhal, D. *et al.*, 1999. "Improved design of "Anytown" distribution network using structured messy genetic algorithms". Urban Water, vol. 1, no. 1, pp. 23–38.

Wu, Z. Y., Boulos, P. F. *et al.*, 2001. "Using genetic algorithms to rehabilitate distribution systems". Journal American Water Works Association, vol. 93, no. 11, pp. 74–85.

10. Responsibility notice

The author(s) is (are) the only responsible for the printed material included in this paper